Amortized Analysis of Fibonacci Heaps

$size(v)$	number of elements in subtree rooted at v
$degree(v)$	degree of node v
$D(n)$	maximum degree of any node in a Fibonacci heap with n elements
$t(H)$	number of trees in Fibonacci heap H
$m(H)$	number of marked nodes in Fibonacci heap H

Potential of a Fibonacci heap H:

$$\Phi(H) = t(H) + 2m(H)$$

	actual cost	$\Delta\Phi$ (upper bound)
INSERT	$O(1)$	1
MELD	$O(1)$	0

INSERT takes constant time and increases the potential by adding a new tree.
MELD takes constant time and does not change the potential.
For both operations, marks are not affected.

	actual cost	$\Delta\Phi$ (upper bound)
DeleteMin	$O(D(n)+t(H))$	$D(n)+1-t(H)$

Inserting the at most $D(n)$ children into the root list takes time $O(D(n))$. CONSOLIDATE takes time proportional to the size of the root list, i.e., $O(D(n)+t(H))$.

The number of trees was $t(H)$ before and is at most $D(n)+1$ afterwards. The number of marked nodes does not increase.

	actual cost	$\Delta\Phi$ (upper bound)
DecreaseKey	$O(c+1)$	$c-2(c-2)=4-c$

where c is the number of cascading cut operations involved in the DecreaseKey operation.

Each cut takes constant time. Except for the last one, each cascading cut clears a mark, whereas the last cut marks a parent if any. Thus the number of marked nodes decreases by $c-2$. The root list increases by c nodes at most.

Theorem:
The maximum degree $D(n)$ of a node in a Fibonacci heap of n elements is in $O(\log n)$.

Lemma:
Let v be a node in a Fibonacci heap with degree k. Let $y_1, y_2, \ldots, y_k$ be the children of v in the order in which they were linked to v, from the earliest to the latest. Then $degree(y_1) \geq 0$ and $degree(y_i) \geq i-2$ for $i = 2, 3, \ldots, k$.

When y_i is linked to v, we have $degree(y_i) = degree(v) \geq i-1$. Since then at most one child was cut.

Let $s_k =$ minimum size of all nodes z with $degree(z) = k$

Then $s_0 = 1$, $s_1 = 2$, $s_2 = 3$ and

Lemma:
For all $k \geq 0$ we have $s_k \ \geq \ \text{FIB}(k+2)$.

Reminder:
$\text{FIB}(0) = 0$, $\text{FIB}(1) = 1$, $\text{FIB}(2) = 1$, and $\text{FIB}(n) = \text{FIB}(n-1) + \text{FIB}(n-2)$ and

$$\text{FIB}(k+2) = 1 + \sum_{i=0}^{k} \text{FIB}(i) = 2 + \sum_{i=2}^{k} \text{FIB}(i)$$

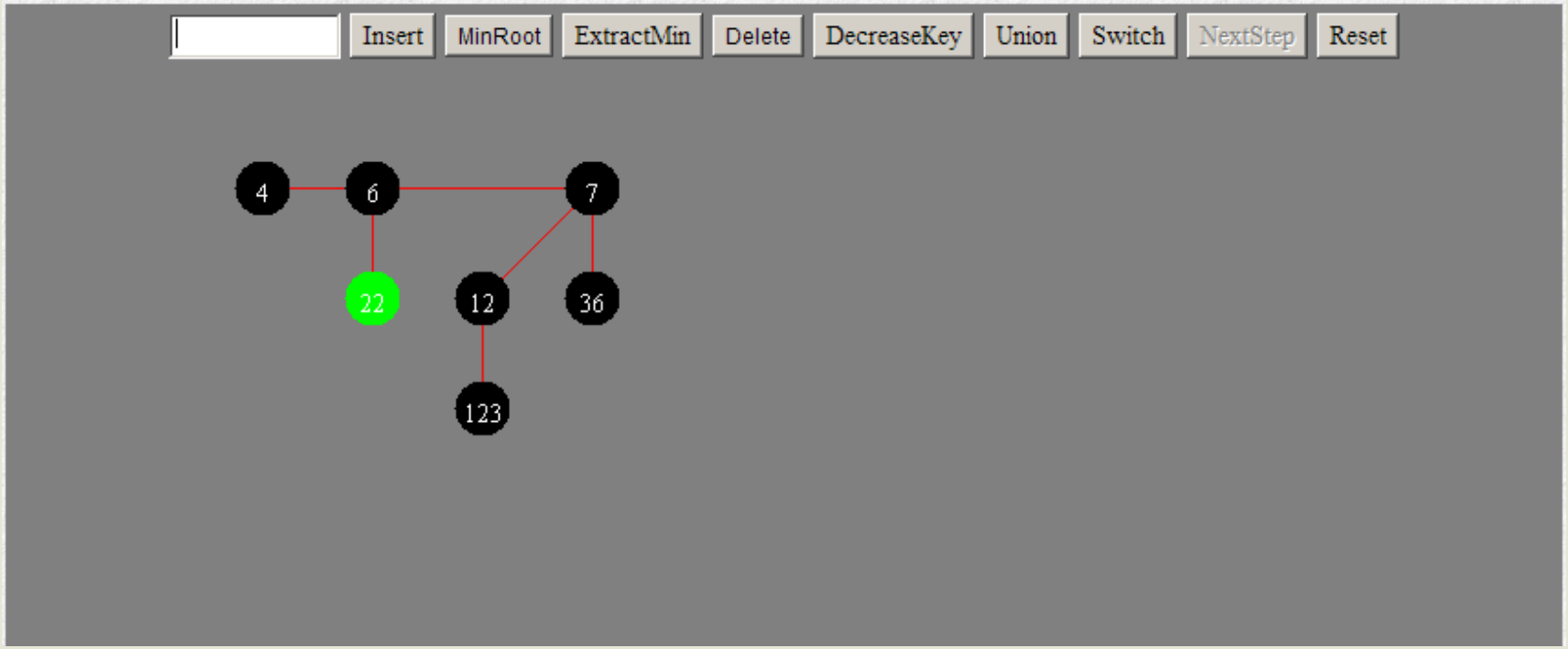
Let v be a node of degree k. Then

$$\begin{aligned}
size(v) &= 1+\sum_{i=1}^{k} size(y_i) = 2+\sum_{i=2}^{k} size(y_i) \\
&\geq 2+\sum_{i=2}^{k} s_{degree(y_i)} \\
&\geq 2+\sum_{i=2}^{k} s_{i-2} \\
&\geq 2+\sum_{i=2}^{k} \text{FIB}(i) = \text{FIB}(k+2)
\end{aligned}$$

The theorem follows since $n \geq size(v) \geq \text{FIB}(k+2) \geq \left(\frac{1+\sqrt{5}}{2}\right)^k$.

Fibonacci Heap

Algorithm in pseudo code and how to use this Applet



Amortized Analysis of Binomial Heaps

INSERT	$O(1)$	amortized
DELETEMIN	$O(\log n)$	worst-case
DELETE	$O(\log n)$	worst-case
DECREASEKEY	$O(\log n)$	worst-case
MELD	$O(1)$	amortized

Insertions and meld operations are analogous to incrementing and adding binary numbers. Define the potential to be the number of trees in the heap plus the lenght of the binary representation of the number of elements in the heap. Regarding the latter, note that we have two numbers before the meld and only one afterwards.